

Taller de Bash Inicial

Juan Pablo Darago - Julian Sackmann - Pablo Antonio

13 de junio de 2014

- 1 Introducción
- 2 Presentandonos
- 3 Conceptos importantes
- 4 Bueno, ¡A practicar!

Introducción

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.
- Marco: Talleres de “cosas útiles”

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.
- Marco: Talleres de “cosas útiles”
 - ComCom!

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.
- Marco: Talleres de “cosas útiles”
 - ComCom!
 - **Comisión** de **Computadores**.

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.
- Marco: Talleres de “cosas útiles”
 - ComCom!
 - **Com**isión de **Com**putadores.
- Primer paso: aprender a usar la consola.

¡Bienvenidos!

- Hoy vamos a dar los primeros pasos con la terminal.
- Marco: Talleres de “cosas útiles”
 - ComCom!
 - **Com**isión de **Com**putadores.
- Primer paso: aprender a usar la consola.
 - Olvidarnos de la GUI por un rato.

¿Qué es la terminal?

- El coso negro con letras verdes.

¿Qué es la terminal?



¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.
 - Recibir una linea que dice que programa correr y como.

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.
 - Recibir una linea que dice que programa correr y como.
 - Correr ese programa.

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.
 - Recibir una linea que dice que programa correr y como.
 - Correr ese programa.
- Interfaz entre el usuario y los otros programas.

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.
 - Recibir una linea que dice que programa correr y como.
 - Correr ese programa.
- Interfaz entre el usuario y los otros programas.
 - A veces parte del sistema operativo (DOS).

¿Que es la terminal?

- Antes de los manejadores de ventanas, existia la terminal.
- Interprete de comandos.
 - Recibir una linea que dice que programa correr y como.
 - Correr ese programa.
- Interfaz entre el usuario y los otros programas.
 - A veces parte del sistema operativo (DOS).
 - ... Idealmente no (UNIX).

Y ahora el que habla se siente



Motivación

- Vamos a hacer una mini-encuesta.

Motivación

- ¿Por qué vamos a ver la terminal si puedo hacer lo mismo con el mouse?

Motivación

- ¿Por qué vamos a ver la terminal si puedo hacer lo mismo con el mouse?
 - Porque usar el mouse es **lento**.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.
 - “Programs should do one thing and one thing only” - Unix Philosophy.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.
 - “Programs should do one thing and one thing only” - Unix Philosophy.
 - Juntarlos para lograr lo que queremos.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.
 - “Programs should do one thing and one thing only” - Unix Philosophy.
 - Juntarlos para lograr lo que queremos.
- Juntar comandos en *scripts*.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.
 - “Programs should do one thing and one thing only” - Unix Philosophy.
 - Juntarlos para lograr lo que queremos.
- Juntar comandos en *scripts*.
 - Una tarea es ahora instantánea y repetible.

¿Para que sirve la terminal?

- La terminal brilla en combinar comandos.
 - “Programs should do one thing and one thing only” - Unix Philosophy.
 - Juntarlos para lograr lo que queremos.
- Juntar comandos en *scripts*.
 - Una tarea es ahora instantánea y repetible.
 - Documentar como se hace una cierta cosa.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.
 - A la red.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.
 - A la red.
 - A la pantalla.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.
 - A la red.
 - A la pantalla.
 - etc.

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.
 - A la red.
 - A la pantalla.
 - etc.
- En Unix, la terminal **es un programa mas.**

¿En dónde corre la terminal?

- La terminal necesita programas para correr.
- Necesita un sistema operativo para proveer a los programas.
 - El acceso al sistema de archivos.
 - A la red.
 - A la pantalla.
 - etc.
- En Unix, la terminal **es un programa mas**.
 - Usa el SO para levantar otros programas.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:
 - Scripteabilidad

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:
 - Scripteabilidad
 - *Autocompletion*.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:
 - Scripteabilidad
 - *Autocompletion*.
 - *Plugins* y extensibilidad.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:
 - Scripteabilidad
 - *Autocompletion*.
 - *Plugins* y extensibilidad.
- Los programas que abren una terminal no son la terminal.

¿Que terminales hay?

- Al ser un programa, hay muchas variantes.
 - Ejemplos: sh, bash, zsh, fish, tcsh, csh, ksh, ...
 - Nosotros vamos a usar bash.
 - Prueben zsh si tienen ganas.
- Variantes:
 - Scripteabilidad
 - *Autocompletion*.
 - *Plugins* y extensibilidad.
- Los programas que abren una terminal no son la terminal.
 - Ejemplo: GNOME Terminal, XTerm, ITerm, ...

Presentandonos

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).
 - Funcionalidad

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).
 - Funcionalidad
- No es tan facil de adaptar para PowerShell o Cygwin.

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).
 - Funcionalidad
- No es tan facil de adaptar para PowerShell o Cygwin.
 - Pero [babun <https://github.com/babun/babun>] hace maravillas

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).
 - Funcionalidad
- No es tan facil de adaptar para PowerShell o Cygwin.
 - Pero [babun <https://github.com/babun/babun>] hace maravillas
- Primero la presentamos y despues vamos a ver conceptos.

¡Empecemos!

- Vamos a usar Bash: es un excelente promedio entre:
 - Masividad (viene preinstalado en casi todos lados: Ubuntu, Debian, OS X).
 - Funcionalidad
- No es tan facil de adaptar para PowerShell o Cygwin.
 - Pero [babun <https://github.com/babun/babun>] hace maravillas
- Primero la presentamos y despues vamos a ver conceptos.
- Una última cosa: *man*.

Demo inicial

¡Con ustedes la terminal!

Conceptos importantes

Filesystem

- Consiste de archivos organizados en directorios.

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raiz, el “/”.

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raiz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raíz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”
- Cada usuario tiene archivos especiales sobre sus preferencias.

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raíz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”
- Cada usuario tiene archivos especiales sobre sus preferencias.
 - en “.bashrc” las preferencias de Bash.

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raíz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”
- Cada usuario tiene archivos especiales sobre sus preferencias.
 - en “.bashrc” las preferencias de Bash.
- Algunos archivos por defecto son ocultos.

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raíz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”
- Cada usuario tiene archivos especiales sobre sus preferencias.
 - en “.bashrc” las preferencias de Bash.
- Algunos archivos por defecto son ocultos.
 - Todo lo que empiece con “.”

Filesystem

- Consiste de archivos organizados en directorios.
- Hay un directorio raíz, el “/”.
- Cada usuario tiene un directorio suyo en el “/home/”
- Cada usuario tiene archivos especiales sobre sus preferencias.
 - en “.bashrc” las preferencias de Bash.
- Algunos archivos por defecto son ocultos.
 - Todo lo que empiece con “.”
- ¡DEMO!

Procesos

- Cuando corremos un programa creamos un proceso.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.
 - Para suspenderlo, forzarlo a terminar, etc.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.
 - Para suspenderlo, forzarlo a terminar, etc.
- Podemos ver los procesos corriendo con *ps*, *top* o *htop*.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.
 - Para suspenderlo, forzarlo a terminar, etc.
- Podemos ver los procesos corriendo con *ps*, *top* o *htop*.
- Podemos enviar una señal a un proceso con *kill*.

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.
 - Para suspenderlo, forzarlo a terminar, etc.
- Podemos ver los procesos corriendo con *ps*, *top* o *htop*.
- Podemos enviar una señal a un proceso con *kill*.
 - Para eso necesitamos su PID (Process ID).

Procesos

- Cuando corremos un programa creamos un proceso.
 - El shell busca el programa y lo corre como proceso.
- El proceso ejecuta hasta que termina o le mandan una señal.
 - Para suspenderlo, forzarlo a terminar, etc.
- Podemos ver los procesos corriendo con *ps*, *top* o *htop*.
- Podemos enviar una señal a un proceso con *kill*.
 - Para eso necesitamos su PID (Process ID).
- ¡DEMO!

Usuarios

- El sistema tiene multiples usuarios.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.
 - Permisos para los demas.

Usuarios

- El sistema tiene múltiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo está mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.
 - Permisos para los demás.
- Los permisos indican si se puede escribir, leer o ejecutar.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.
 - Permisos para los demas.
- Los permisos indican si se puede escribir, leer o ejecutar.
- El owner y group se cambia con *chown* y *chmod*.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.
 - Permisos para los demas.
- Los permisos indican si se puede escribir, leer o ejecutar.
- El owner y group se cambia con *chown* y *chmod*.
- Los permisos con *chmod*.

Usuarios

- El sistema tiene multiples usuarios.
- Cada usuario es dueño de recursos:
 - Sus archivos.
 - Sus procesos.
- El superusuario *root* puede hacer lo que quiera.
- El acceso a un archivo esta mediado por sus permisos.
 - Permisos para el *owner*.
 - Permisos para el *group* del *owner*.
 - Permisos para los demas.
- Los permisos indican si se puede escribir, leer o ejecutar.
- El owner y group se cambia con *chown* y *chmod*.
- Los permisos con *chmod*.
- ¡DEMO!

Editores

- Hay editores de texto solo para la terminal.

Editores

- Hay editores de texto solo para la terminal.
- No necesitan cliente de terminal, se pueden usar por ssh.

Editores

- Hay editores de texto solo para la terminal.
- No necesitan cliente de terminal, se pueden usar por ssh.
- Lo mas populares son:

Editores

- Hay editores de texto solo para la terminal.
- No necesitan cliente de terminal, se pueden usar por ssh.
- Lo mas populares son:
 - *nano*: Util y productivo, facil de usar.

Editores

- Hay editores de texto solo para la terminal.
- No necesitan cliente de terminal, se pueden usar por ssh.
- Lo mas populares son:
 - *nano*: Util y productivo, facil de usar.
 - *vim*: Modal, muy productivo, curva de aprendizaje media.

Editores

- Hay editores de texto solo para la terminal.
- No necesitan cliente de terminal, se pueden usar por ssh.
- Lo mas populares son:
 - *nano*: Util y productivo, facil de usar.
 - *vim*: Modal, muy productivo, curva de aprendizaje media.
 - *emacs*: Muy poderoso, curva de aprendizaje alta.

Maquinas remotas

- Podemos conectarnos a maquinas externas.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.
- Pueden conectarse a las maquinas de los labos desde afuera con *ssh*.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.
- Pueden conectarse a las maquinas de los labos desde afuera con *ssh*.
 - Usando el servidor *milagro*.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.
- Pueden conectarse a las maquinas de los labos desde afuera con *ssh*.
 - Usando el servidor *milagro*.
 - ¡DEMO!

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.
- Pueden conectarse a las maquinas de los labos desde afuera con *ssh*.
 - Usando el servidor *milagro*.
 - ¡DEMO!
- También pueden copiar archivos por SCP.

Maquinas remotas

- Podemos conectarnos a maquinas externas.
- Usando para eso *ssh* (Secure Shell).
 - Nuestro programa *ssh* habla con otro programa (*sshd*) en la otra maquina que interpreta los comandos.
 - Simplemente ejecuta *bash* y manda las cosas por la red.
 - Encriptado y muy seguro: no pueden mirar lo que hacemos.
- Pueden conectarse a las maquinas de los labos desde afuera con *ssh*.
 - Usando el servidor *milagro*.
 - ¡DEMO!
- También pueden copiar archivos por SCP.
 - ¡DEMO!

Un par de tips

- Aliasing

Un par de tips

- Aliasing
- TMUX

Un par de tips

- Aliasing
- TMUX
- *man pages*

Un par de tips

- Aliasing
- TMUX
- *man pages*
 - Ya se que lo dijimos, pero ¡posta es importante!

Bueno, ¡A practicar!

Ejercitación.

- Ahora vamos a hacer ejercicios prácticos.

Ejercitación.

- Ahora vamos a hacer ejercicios prácticos.
- No estan subidos a ningun lado:

Ejercitación.

- Ahora vamos a hacer ejercicios prácticos.
- No estan subidos a ningun lado:
 - ¡Tienen que copiar los comandos a mano!

Ejercitación.

- Ahora vamos a hacer ejercicios prácticos.
- No estan subidos a ningun lado:
 - ¡Tienen que copiar los comandos a mano!
- Para algunos ejercicios van a necesitar un zip de cubawiki.

Ejercitación.

- Ahora vamos a hacer ejercicios prácticos.
- No estan subidos a ningun lado:
 - ¡Tienen que copiar los comandos a mano!
- Para algunos ejercicios van a necesitar un zip de cubawiki.
 - Cualquier cosa este es el link:
<http://cubawiki.com.ar/tallerBash.tar.gz>

Pedir ayuda:

- Si necesitan ayuda pueden:

Pedir ayuda:

```
NPM-JSON(1)                                     NPM-JSON(1)

NAME
  npm-json — Specifics of npm's package.json handling

DESCRIPTION
  This document is all you need to know about what's required in your
  package.json file. It must be actual JSON, not just a JavaScript
  object literal.

  A lot of the behavior described in this document is affected by the
  config settings described in npm help config.

DEFAULT VALUES
  npm will default some values based on package contents.

  o "scripts": {"start": "node server.js"}

    If there is a server.js file in the root of your package, then npm
    will default the start command to node server.js.

  o "scripts":{"preinstall": "node-waf clean || true; node-waf config-
```

:|

Pedir ayuda:



Pedir ayuda:



Pedir ayuda:

- Preguntarnos a nosotros!

Pedir ayuda:

- Preguntarnos a nosotros!
 - O a cualquiera de los chicos que generosamente se ofrecieron a ayudarnos.

Pedir ayuda:

- Preguntarnos a nosotros!
 - O a cualquiera de los chicos que generosamente se ofrecieron a ayudarnos.
- Aunque no sabemos tanto como Google. . .

Pedir ayuda:

- Preguntarnos a nosotros!
 - O a cualquiera de los chicos que generosamente se ofrecieron a ayudarnos.
- Aunque no sabemos tanto como Google. . .
- . . . ni como las man pages. . .

Pedir ayuda:

- Preguntarnos a nosotros!
 - O a cualquiera de los chicos que generosamente se ofrecieron a ayudarnos.
- Aunque no sabemos tanto como Google. . .
- . . . ni como las man pages. . .
- . . . pero seguro que dirigimos fútbol mejor que Nishimura.

A trabajar

- ¡Arranquemos!